AD-A101 694 PURDUE UNIV LAFAYETTE IN DEPT OF COMPUTER SCIENCES F/G 9/2
SYSTEMATIC STUDY OF COBOL PROGRAMMING.(U)
JUN 81 S D CONTE, V Y SHEN DAAG29-79-C-0172
UNCLASSIFIED ARO-16400.4-A-EL NL

LEVEL II (19) (12)

(18) ARO 16400.4-A-EL

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD A101694

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: DRXRO-IP-L-16400-A-EL	2. GOVT ACCESSION NO.: AD-A101694
3. RECIPIENT'S CATALOG NUMBER:	
4. TITLE (and Subtitle): Systematic Study of COBOL Programming	5. TYPE OF REPORT & PERIOD COVERED: Final 9/19/79 to 3/8/81
	6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): S.D. Conte - V.Y. Shen	8. CONTRACT OR GRANT NUMBER(s): DAAG29-79-C-0172
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Division of Sponsored Programs, Purdue University, W. Lafayette, IN 47907	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:
11. CONTROLLING OFFICE NAME AND ADDRESS: U. S. Army Research Office, Post Office Box 12211, Research Triangle Park, NC 27709	12. REPORT DATE: 6-4-81 13. NUMBER OF PAGES: 4
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): AIRMICS	15. SECURITY CLASS. (of this report): Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

DTIC ELECTE JUL 21 1981 S D

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

NA

18. SUPPLEMENTARY NOTES

The view, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Software Science, Cobol Analyzer, Software Engineering, Modularity, Effort Estimation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report studies the applicability of the Theory of Software Science to COBOL Programs. A COBOL analyzer was written to produce the most important Software Science metrics. The analyzer was used to study several COBOL Databases. The analyzer has been distributed to AIRMICS and to several other organizations including Ohio State University, Jet Propulsion Labs, CINCOM INC., and the University of Trondheim, Norway.

(continued on back)

DTIC FILE COPY

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

81 7 17 103

UNCLASSIFIED

(20 continued)

Using the analyzer we investigated the Software Science Effort Estimator and compared this with estimator based on lines of code and on "cyclometric complexity." Our experiments show that the software science effort estimator works reasonably well only if a program is properly modularized and if inter-module factors are included. Published reports on our findings are available on request.

We also studied the language level hypothesis of software science in a COBOL environment. While the study shows that the mean language level for COBOL falls between those for FORTRAN and PL/I, the study also shows that language level is heavily dependent on the size of the program and that it fluctuates too much to be considered "constant" as the Halstead theory postulates.

Final Report

Systematic Study of COBOL Programming

S. D. Conte

V. Y. Shen

Computer Sciences Department

Purdue University

W. Lafayette, IN 47907

Period covered by report: 09/09/79-03/08/81

ARO project number: P-16400-A-EL

Contract number: DAAG29-79-C-0172

Scientific Personnel Supported by this Project:

Alka Agrawal

Dennis Cok (M.Sc. 1980)

Stephen M. Thebaut

Scott N. Woodfield (Ph.D. 1980)

1. The Problem Studied

The purpose of this research was to study the applicability of the theory of Software Science in managing the development of COBOL programs. Since Software Science was a theory in its infancy, we hoped that through this study we could obtain evidences supporting the theory or make refinements to the theory.

2. Summary of Important Results

During the contract period we have developed an analyzer for COBOL programs, gained additional insight in modularized program development, and discovered the limitations on the language level hypothesis of Software Science. These three accomplishments are described below:

2.1. The COBOL Analyzer

The necessary tool for systematically studying COBOL programs is an automatic analyzer. It was completed in early 1980. We used it to analyze a variety of programs and the preliminary results were reported in [1]. The major findings were that the Software Science length equation works as well in COBOL programs as in other languages previously analyzed; that the effort equation appeared to work for one single-programmer, single-module program; and that the language level equation did not provide stable results. To date the analyzer has been distributed to the following institutions for research purposes:

AIRMICS

The Ohio State University

Jet Propulsion Laboratory

Cincom, Inc.

The University of Trondheim, Norway

Mr. Wilhem Otnes of the University of Trondheim has conducted an elaborate experiment using the COBOL analyzer. He wrote a report which generally confirmed our findings [2].

2.2. Modularity Factors in Effort Estimation

Effective software management requires accurate estimation of the programming effort. The "lines of code" measure, the "cyclomatic complexity" measure, and the "effort" measure of Software Science have all been proposed to be important factors in programming effort estimation. We have conducted experiments which showed that neither measure was adequate in estimating effort for single-programmer, multi-module programs unless the modularization and module-interconnection factors were included [3]. This important extension of the Software Science theory has yielded two publications [4,5]. A third report on the research has been submitted [6].

2.3. The Language Level

A hypothesis of Software Science is that the power of a programming language can be quantified by the "language level" equation under certain assumptions. Previously reported studied shows that the equation yields numbers that rank languages such as PL/1, Algol, FORTRAN, and assembly language in the desired order. However, the numbers reported were based on hand-analysis of small sets of programs. The development of automatic analyzers enabled us to analyze hundreds of programs written in COBOL, FORTRAN, Pascal, PL/S, and assembly language. The language level based on Software Science theory does not remain constant. It appears to have a strong dependency on program length. The results of our analysis are reported in [7].

3. List of Publications

[1] Shen, V. Y. and Dunsmore, H. E., "A Software Science Analysis of COBOL Programs", CSD-TR-348, Purdue University, August 1980. Submitted to *IEEE Trans. Software Engineering*.

[2] Otnes, W., "Quantitative Analysis of Software Projects", Tech. Rep. No. 15/80, The University of Trondheim, The Norwegian Institute of Technology, Dec. 1980.

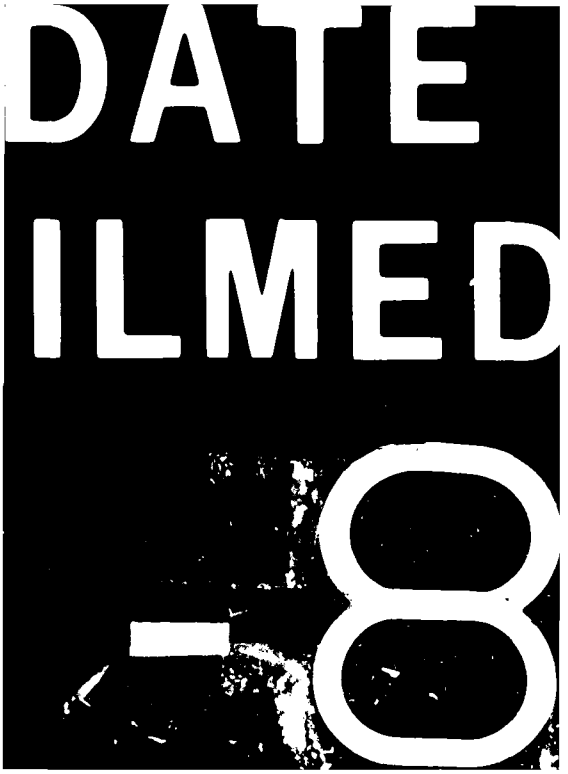
[3] Woodfield, S. N., *Enhanced Effort Estimation by Extending Basic Programming Models to Include Modularity Factors*, Ph.D. Thesis, Computer Sciences Department, Purdue University, December 1980.

[4] Woodfield, S. N., Dunsmore, H. E., and Shen, V. Y., "The Effect of Modularization and Comments on Program Comprehension", *Proc. 5th Int. Conf. on Software Engineering*, pp. 215-223, March 1981.

[5] Woodfield, S. N., Shen, V. Y., and Dunsmore H. E., "A Study of Several Metrics for Programming Effort", *Proc. 1st ACM Symposium on Software Engineering*, Pingree, Colorado, June 1981. *To appear in J. Systems and Software.*

[6] Woodfield, S. N., Shen, V. Y., and Dunsmore, H. E., "A Module Interconnection Complexity Measure", submitted to *IEEE Trans. Software Engineering.*

[7] Conte, S. D., "Investigation of Language Level in Software Science", Preliminary Report, Computer Sciences Department, Purdue University, October 1980.


DATE
ILMED
-8